Algorithmica (2001) 31: 79–113
DOI: 10.1007/s00453-001-0040-8

Algorithmica

Inapproximability Results for Guarding Polygons and Terrains[1]

S. Eidenbenz,[2] C. Stamm,[2] and P. Widmayer[2]

Abstract. Past research on art gallery problems has concentrated almost exclusively on bounds on the numbers of guards needed in the worst case in various settings. On the complexity side, fewer results are available. For instance, it has long been known that placing a smallest number of guards for a given input polygon is *NP*-hard. In this paper we initiate the study of the approximability of several types of art gallery problems.

Motivated by a practical problem, we study the approximation properties of the three art gallery problems VERTEX GUARD, EDGE GUARD, and POINT GUARD. We prove that if the input polygon has no holes, there is a constant $\delta > 0$ such that no polynomial time algorithm can achieve an approximation ratio of $1+\delta$, for each of these guard problems. We obtain these results by proposing gap-preserving reductions from 5-OCCURRENCE-MAX-3-SAT.

We also prove that if the input polygons are allowed to contain holes, then these problems cannot be approximated by a polynomial time algorithm with ratio $((1-\varepsilon)/12)\ln n$ for any $\varepsilon > 0$, unless $NP \subseteq TIME(n^{O(\log\log n)})$, where n is the number of vertices of the input polygon. We obtain these results by proposing gap-preserving reductions from the SET COVER problem.

We show that this inapproximability for the POINT GUARD problem for polygons with holes carries over to the problem of covering a 2.5-dimensional terrain with a minimum number of guards that are placed at a certain fixed height above the terrain. The same holds if the guards may be placed on the terrain itself.

Key Words. Art gallery, Visibility problems, Inapproximability, Gap-preserving reductions, Terrains, Telecommunications.

1. Introduction

1.1. *Motivation*. Triggered by the deregulation of the telecommunications markets all over the world, and recently especially in Europe, companies are planning to set up networks for wireless communication. Typically, such a network consists of transmission stations (antennas) that receive and send signals. The set of antennas needs to cover a specific geographic region in its entirety. Putting up antennas is very costly, and hence mobile phone companies aim at placing a minimum number of antennas that cover a given region. Since the traditional way of erecting antenna towers on the ground suffers from a number of obvious disadvantages, a novel approach is to put antennas up in the air: balloons float at a certain fixed height and are held in geo-stationary position. The

[1] This work was partially supported by the Swiss National Science Foundation. Preliminary versions of parts of this work have been reported in [9]–[11].

[2] Institute for Theoretical Computer Science, ETH Zentrum, 8092 Zürich, Switzerland. {eidenben,stamm, widmayer}@inf.ethz.ch.

Received September 22, 1999; revised December 5, 1999. Communicated by B. Chazelle.
Online publication June 11, 2001.

problem of finding a smallest set of balloons at a height of 20 km above sea level that cover Switzerland is indeed the practical motivation behind our theoretical study.

1.2. *The Problem*. In the abstract version of this problem we are given a *terrain*, described as a finite set of points in the plane together with a triangulation, and a height value associated with each point (this is also called a *triangulated irregular network* (*TIN*), see, e.g., [16]). We are further given a desired antenna *height* that is above the highest point of the terrain. The objective is to find a smallest set of *antenna points* at the given height that *covers* the given terrain, that is, each point on the terrain must be visible from at least one of the antenna points. Visibility is defined on the basis of straight *lines-of-sight*: two points above the terrain are mutually visible if their connecting straight line segment runs entirely above (or on) the terrain. This concept models simple electromagnetic wave propagation at high frequencies (GHz), ignoring effects such as reflection and refraction.

1.3. *Related Problems*. The described terrain covering problem can be seen to belong to quite a large family of geometric covering and guarding problems that have been studied for more than two decades. Legend has it that during a conference in 1976, Victor Klee started the study by posing the following problem, which today is known as the *original art gallery problem*: How many guards are needed to see every point in the interior of an art gallery? In the abstract version of this problem, the input is a simple polygon in the plane, representing the floorplan of the art gallery, and visibility is of course limited to the interior of the polygon. The variations of this polygon guarding problem that have been investigated can be classified as to where the guards may be positioned (anywhere, or in any one of a few distinguished locations), what kind of guards are to be used (single points versus sets of points, such as line segments, and guards in stationary positions versus mobile guards), whether only the boundary or all of the interior of the polygon must be guarded, what the assumptions are on the input polygons (such as being simple or orthogonal). Many upper and lower bounds on the number of necessary guards are known for specific settings, while comparatively few papers study the computational complexity of finding good positions for the guards, given a polygon. For more details, see any of several surveys on the general topic of art galleries [18], [19], [23], [25]. In this section we only briefly summarize some relevant previous results and give pointers to the literature; we give the exact definitions of the problems we study in Section 2. For an up-to-date extensive survey on the state of the art, consult [25].

As to the computational complexity of polygon guarding problems, it is known that the problem of covering a polygon with a minimum number of convex polygons (that may overlap) is *NP*-hard for input polygons with holes [20], and also for polygons without holes [8]. The POINT GUARD problem asks for a smallest set of points that together see a given polygon in its entirety; it is equivalent to the problem of covering a polygon with a minimum number of star-shaped polygons, and this is also *NP*-hard for input polygons with [20] and without [17] holes. The two problems VERTEX GUARD and EDGE GUARD (where the guards that together see a given polygon in its entirety are constrained to be vertices and edges of the polygon, respectively) are *NP*-hard even for input polygons without holes [17]. Approximation algorithms for

VERTEX GUARD and EDGE GUARD which achieve approximation ratios of $O(\log n)$ are also known [13]. The approximation algorithms work for polygons with and without holes.

1.4. *Aims and Scope of this Paper.* Urrutia points out that approximability results for polygon guarding problems are dearly needed [25]. The approximability of a problem is interesting from both sides, the upper bound and the lower bound. Upper bounds come from approximation algorithms that achieve a certain approximation ratio, as has been done for VERTEX GUARD and EDGE GUARD [13]. Lower bounds on the approximation ratio are the subject of this paper.

In Section 2 we give exact definitions of the problems we study. In Section 3 we obtain results for VERTEX GUARD, EDGE GUARD, and POINT GUARD for input polygons without holes that prove that for each of these problems there is a constant $\delta > 0$ such that no polynomial time algorithm can achieve an approximation ratio of $1 + \delta$. We obtain these results by proposing gap-preserving reductions [2] from 5-OCCURRENCE-MAX-3-SAT that are based on reductions originally used to prove the *NP*-hardness of these problems [17].

We prove in Section 4 that if the input polygons are allowed to contain holes, then these problems cannot be approximated by a polynomial time algorithm with ratio $(1-\varepsilon)\frac{1}{12}\ln n$ for any $\varepsilon > 0$, unless $NP \subseteq TIME(n^{O(\log\log n)})$, where n is the number of vertices of the input polygon. We obtain these results by proposing gap-preserving reductions from the SET COVER problem (i.e., a restricted version of SET COVER, which corresponds to DOMINATING SET).

We show in Section 5 that our proof for the POINT GUARD problem for polygons with holes carries over to the problem of covering a 2.5-dimensional terrain with a minimum number of guards that are placed either at a certain fixed height above the terrain or on the terrain itself.

After presenting all of these inapproximability results, we summarize the approximability results known for these problems in Section 6. Finally, we show which of our results are optimum and which results might be improved and draw some conclusions in Section 7.

2. Preliminaries

DEFINITION 1. A *polygonal chain* P is an ordered sequence of points $p_1, \ldots, p_n, n \geq 3$, in the plane, called the *vertices* of P, together with the set of line segments joining p_i to p_{i+1}, $i = 1, \ldots n-1$, called the *edges* of P. A polygonal chain is called *closed* if $p_1 = p_n$; otherwise it is called *open*. For a closed polygonal chain, we sometimes refrain from repeating the first vertex p_1, and we end the chain with p_{n-1}.

DEFINITION 2. A polygonal chain is called *simple* if the only intersections of edges are those at common endpoints of consecutive edges. A simple, closed polygonal chain P divides the plane into two regions, the *interior* and the *exterior* of P, where the exterior is the unbounded region and the interior is the bounded region (it does not contain a line or even a half-line).

DEFINITION 3. The interior of a simple, closed polygonal chain P, together with P, is called a *simple polygon without holes*. Its *boundary* δP is just P. For simplicity, the polygon is denoted by P as well.

DEFINITION 4. A *polygon* is the union of a finite number of simple polygons. A polygon P is called *connected*, if any two points of P can be joined by a polygonal chain that belongs to P. A connected polygon P is called *simply connected* if every polygonal chain between two boundary points that does not pass through any other boundary point divides P. A connected polygon that is not simply connected is called a *simple polygon with holes*.

Note that a simple polygon with holes P can be represented by a finite number k of polygonal chains $P_1, \ldots, P_k$ that represent its boundary, where P_1 is the *outer* boundary of the polygon, and the P_i, for $i = 2, \ldots, k$, are the boundaries of the *holes*. For this representation to work, we require that $P_i \subseteq P_1$ for $i = 2, \ldots, k$ and that $P_i \cap P_j = \emptyset$ for $i \neq j$ and $i, j = 2, \ldots, k$. The interior of P is the set difference between P_1 and the interiors of $P_2, \ldots, P_k$ viewed as simple polygons without holes. Note that the boundaries of $P_2, \ldots, P_k$ belong to P.

Since we only deal with connected polygons in the following, we use the term polygon for a connected polygon, with or without holes.

Among the multitude of notions for *visibility* between two points in a polygon, we use the following:

DEFINITION 5. Let P be a polygon, and let A and B be points belonging to P. Points A and B are mutually *visible* with respect to P, if the straight line segment connecting A and B belongs to P. We also say that A and B *see* each other, that A is visible from B, and that A sees B. For a set Q and a set S of points of P, we say that Q is visible from S if for each point $q \in Q$ there is a point $s \in S$ that sees q.

Note that visibility is symmetric for single points, but not for sets of points: while a set Q may be visible from a set S of points with respect to a polygon P, it may not be true that S is visible from Q. We therefore call Q the set of guarded points and S the set of guard points (or simply guards).

We now define the problems we are studying.

DEFINITION 6. Let P be a simple polygon without holes. The problem VERTEX GUARD (VG) is the problem of finding a minimum subset S of (the set of) vertices of P such that the boundary of P is visible from S. The vertices in S are called *vertex guards*.

Note that, as usual, a minimum subset of a set denotes a subset of smallest cardinality among all candidate subsets.

DEFINITION 7. Let P be a simple polygon without holes. The problem EDGE GUARD (EG) is the problem of finding a minimum subset S of edges of P such that the boundary of P is visible from the points in S. The edges in S are called *edge guards*.

DEFINITION 8. Let P be a simple polygon without holes. The problem POINT GUARD (PG) is the problem of finding a minimum set S of points belonging to P such that the boundary of P is visible from S. The points in S are called *point guards*.

VG, EG, and PG can also be defined such that all of the interior (and not only the boundary) of the input polygon P must be visible from at least one guard (see [17]). We denote the corresponding problems by VGI, EGI, PGI, where the letter "I" stands for "interior." Finally, we can define these problems for input polygons with (instead of without) holes. We denote this by adding a letter "H," if we allow the input polygons to contain holes, i.e., VGH, EGH, PGH, VGIH, EGIH, PGIH as opposed to VG, EG, PG, VGI, EGI, PGI.

We can define similar problems when the input structure is a *terrain* rather than a polygon.

DEFINITION 9. A *terrain* T is a two-dimensional surface in three-dimensional space, represented as a finite set of vertices in the plane, together with a triangulation of their planar convex hull, and a height value associated with each vertex. By a linear interpolation in between vertices, this representation defines a bivariate, continuous function. The corresponding surface in space is also called the 2.5-dimensional terrain. A terrain divides three-dimensional space into two subspaces, i.e., a space *above* and a space *below* the terrain, in the obvious way.

For simplicity, we describe the terrain problems in the Cartesian x–y–z space, where the z-value denotes the height of terrain points.

DEFINITION 10. Let T be a terrain, and let A and B be two points in space above or on T. Point A is *visible* from point B with respect to T if the straight line segment connecting A and B is entirely on or above T.

For antennas, this definition does not model all aspects of electromagnetic wave propagation exactly, since, e.g., the signal of an antenna gets weaker as it propagates, and the signal is reflected on a rocky wall. However, for the practical problem that motivates this study, the straight *line-of-sight* (LOS) approach provides a satisfactory approximation of reality.

For a terrain, we consider the following problems:

DEFINITION 11. Let T be a terrain. The problem VERTEX GUARD ON TERRAIN (VGT) is the problem of finding a minimum subset S of vertices of T such that T is visible from S.

DEFINITION 12. Let T be a terrain. The problem POINT GUARD ON TERRAIN (PGT) is the problem of finding a minimum set S of points on T such that T is visible from S.

DEFINITION 13. Let T be a terrain, and let h be a height value, such that the plane $z = h$ lies entirely above (or partially on) T. The problem GUARDS AT FIXED HEIGHT OVER TERRAIN (FHT) is the problem of finding a minimum set S of points in space at height h such that T is visible from S.

We now define three additional, similar terrain guarding problems. The reason is that our inapproximability results for guarding terrains will be formulated for terrain problem versions with the additional restriction that each triangle in the triangulation of T must be visible from a single point in the guard set S; that is, guards are not allowed cooperatively to see a triangle in T's triangulation, contrary to the problem versions above. We denote these problem versions with an extra letter "R" that stands for "restricted." Hence, we get terrain guarding problems VGTR, PGTR, and FHTR.

We prove our inapproximability results by proposing reductions from the following problems.

DEFINITION 14. Let Φ be a boolean formula given in conjunctive normal form, with each clause consisting of at most three literals and with each variable appearing in at most five clauses. The problem 5-OCCURRENCE-MAX-3-SAT consists of finding a truth assignment for the variables of Φ that satisfies as many clauses as possible.

5-OCCURRENCE-MAX-3-SAT is *MAXSNP*-complete [21], which means that there is a constant $\delta > 0$ such that 5-OCCURRENCE-MAX-3-SAT cannot be approximated by a polynomial time algorithm with a ratio $1 + \delta$ unless $NP = P$. (It also means that there exists a polynomial time approximation algorithm for the problem that achieves a constant ratio.) Let *APX* be the class of optimization problems that can be approximated by polynomial time algorithms with a constant ratio (see [6], [7], [15], and [24] for details on the relationship of these two complexity classes). 5-OCCURRENCE-MAX-3-SAT is APX-complete as well.

DEFINITION 15. Let $E = \{e_1, \ldots, e_n\}$ be a finite set (called universe) of elements, and let $S = \{s_1, \ldots, s_m\}$ be a collection of subsets of E, i.e., $s_j \subseteq E$ for $1 \leq j \leq m$. The problem SET COVER (SC) is the problem of finding a minimum subset $S' \subseteq S$ such that every element $e_i \in E$, $1 \leq i \leq n$, belongs to at least one subset in S'. For ease of discussion, let the elements in E and the subsets in S have an arbitrary, but fixed, order, denoted by the index.

DEFINITION 16. Let $G = (V, E)$ be an undirected graph with n vertices $V = \{v_1, \ldots, v_n\}$ and edges E. The problem DOMINATING SET consists of finding a minimum set S' of vertices such that each vertex $v_i \in V$ has at least one neighboring vertex in S', i.e., for each vertex $v_i \in V$, there exists a vertex $v_j \in S'$ with $(v_i, v_j) \in E$.

SET COVER and DOMINATING SET can be approximated in polynomial time with a ratio of $1 + \ln n$ by a simple greedy algorithm [4], [14], but it cannot be approximated by any polynomial time algorithm with a ratio of $(1 - \varepsilon) \ln n$, for any $\varepsilon > 0$ unless $NP \subseteq TIME(n^{O(\log \log n)})$, where n is the number of elements for SET COVER and the number of vertices in the graph for DOMINATING SET [4], [12].

3. Inapproximability Results for Guarding Polygons without Holes. In this section we propose a reduction from 5-OCCURRENCE-MAX-3-SAT to PG, analyze it, and show

that it is gap-preserving. This implies that PG is *APX*-hard. We also show that *APX*-hardness follows for the problems VG, EG, VGI, EGI, and PGI.

3.1. *Construction of the Reduction.* We present the construction for PG. Suppose we are given an instance I of 5-OCCURRENCE-MAX-3-SAT. Let I consist of n variables $x_1, \ldots, x_n$ and of $m \leq \frac{5}{3}n$ clauses $c_1, \ldots, c_m$. Taking instance I as input, we construct a polygon P, which is an instance I' of PG.

Overview. The polygon P contains six different kinds of basic units, called patterns. These are called *literal*, *clause*, *variable*, *ear*, *body*, and *spike patterns*. Each pattern is a polygon, which will be part of the final polygon P. We obtain the final polygon by taking the union of all patterns. Each pattern (except for the body pattern) contains a *distinguished tuple*.

DEFINITION 17. A *distinguished tuple* (p_i, p_j, p_k, p_l) of a pattern is formed by the four vertices p_i, p_j, p_k, p_l of the pattern with the following properties:

- p_i and p_j are neighboring vertices (i.e., $j = i + 1$ or $j = i - 1$).
- Any guard that sees an arbitrarily small part of the edge from p_i to p_j, which includes vertex p_i, must lie inside or on the boundary of the pattern, even if we consider the edges of the pattern, which form the path between the vertices p_k and p_l (not taking the route that includes vertex p_i), to be transparent.

As an example, consider the polygon in Figure 2, in which (q_6, q_5, q_8, q_1) is a distinguished tuple. The fact that the edge from q_1 to q_8 is transparent is indicated by a dashed line. Polygon components will be composed by attaching them to each other at transparent edges; these edges will, hence, disappear in the composition, and their transparency indicates just this.

DEFINITION 18. Let (p_i, p_j, p_k, p_l) be a distinguished tuple of a pattern. Then an arbitrarily small part of the edge from p_i to p_j starting at p_i and going toward p_j is called a *distinguished arrow*.

Distinguished tuples and distinguished arrows will help us define an algorithm that obtains a truth assignment for the variables of I, if it is given a solution (i.e., a set of guards) of I'.

The reduction works as follows: For each literal, we construct a literal pattern, each of which contains a vertex T^{lit} and a vertex F^{lit}, which corresponds to the truth value of the literal, if a guard sits there. Three literal patterns form a clause pattern in such a way that the clause pattern can only be guarded by a minimum number of guards, if at least one literal in the clause is true. We construct a variable pattern for each variable, which contains a vertex T^{var} and a vertex F^{var}, which corresponds to the truth value of the variable. Finally, spike patterns are used to connect variable and literal patterns in such a way that a minimum number of guards is only possible, if the truth values are assigned consistently.

We first introduce the literal, clause, variable, ear, and body patterns. We then show how these patterns are put together, and finally we define spike patterns.

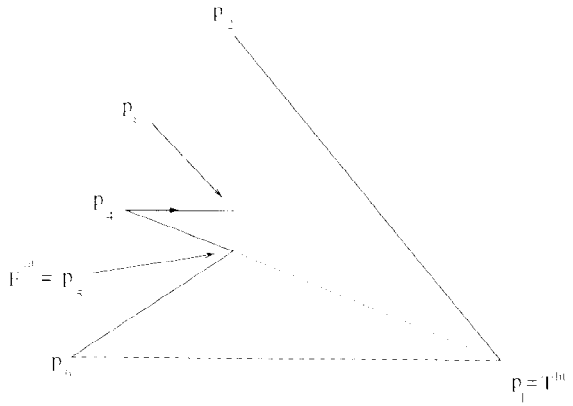

Fig. 1. Literal pattern.

Literal pattern. Let $l_j(c_i)$, for $j = 1, \ldots, 3$ and $i = 1, \ldots, m$, denote the jth literal of the ith clause. Note that $l_j(c_i) = x_k$ or $l_j(c_i) = \neg x_k$ for some $k = 1, \ldots, n$. For each literal $l_j(c_i)$, we construct a literal pattern as shown in Figure 1. The literal pattern is the polygon defined by the points $p_1(l_j(c_i)), \ldots, p_6(l_j(c_i))$, given in counterclockwise order as shown in Figure 1. Whenever it is clear which literal we are talking about, we will denote vertex $p_k(l_j(c_i))$ simply by p_k, omitting the argument, as done in Figure 1. The edge from p_6 to p_1 is not part of the final polygon, but serves as an interface to the outside of the literal pattern. We will lose this edge when we form the union of the literal pattern with a clause pattern. As before, the transparent edge from p_1 to p_6 is drawn as a dashed line. All other edges in the literal pattern are part of the final polygon. The points p_4, p_5, p_1 are collinear. Note that a guard at point p_1 or point p_5 sees all of the interior of the literal pattern. The final construction will be such that a guard at point p_1 implies that the literal is true and a guard at point p_5 implies that the literal is false. We, therefore, call point $p_1(l_j(c_i))$ simply $T^{\text{lit}}(l_j(c_i))$; similarly, $p_5(l_j(c_i))$ is called $F^{\text{lit}}(l_j(c_i))$. Note that (p_4, p_3, p_1, p_6) is a distinguished tuple. The distinguished arrow (p_4, p_3) is marked by an arrow in Figure 1.

Clause pattern. For every clause c_i, we construct a clause pattern as shown in Figure 2. The clause pattern for c_i is the polygon defined by the vertices $q_1(c_i), \ldots, q_8(c_i)$. Vertices q_6, q_7, q_2, and q_3 are collinear. The tuple (q_6, q_5, q_1, q_8) is a distinguished tuple.

We form the union of the clause pattern of clause c_i and the three literal patterns $l_1(c_i)$, $l_2(c_i)$, and $l_1(c_i)$ as indicated in Figure 3. Note that this is done in such a way that a guard at vertex T^{lit} of any of the three literals sees the distinguished arrow of the clause

Fig. 2. Clause pattern.

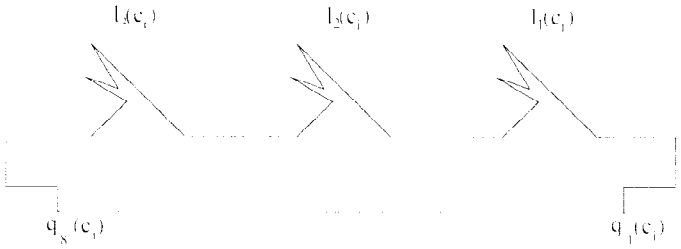

Fig. 3. Union of clause pattern and literal patterns.

pattern, while a guard at a vertex F^{lit} of any literal pattern cannot see the distinguished arrow of the clause pattern.

Variable pattern. For every variable x_k, we construct a variable pattern as shown in Figure 4. The variable pattern is the polygon defined by the vertices $r_1(x_k), \ldots, r_{11}(x_k)$. We call the polygon defined by the vertices r_1, r_2, r_3, r_4 the *TRUE leg* of the variable pattern and the polygon defined by r_5, r_6, r_7, r_{11} the *FALSE leg* of the variable pattern. The vertices r_9, r_{10}, r_1 are collinear, and so are vertices r_7, r_8, r_{10}, r_{11}, and also vertices r_1, r_4, r_5, r_8. The shape of the variable pattern can be changed slightly (as will be done in the final construction), as long as the collinearities are maintained. The tuple (r_9, r_8, r_1, r_{11}) is a distinguished tuple.

In the final polygon it will turn out that a guard sits at point r_1 if the variable is assigned the value true, and it sits at point r_5 if the variable is false. Therefore, we define $T^{\text{var}}(x_k) := r_1(x_k)$ and $F^{\text{var}}(x_k) := r_5(x_k)$.

Ear pattern. The ear pattern is necessary for technical reasons. Its use will become evident in the analysis of the reduction. An ear pattern is the same as a literal pattern. However, it is not associated with any literal. We use the same numbering as for the literal pattern and denote the vertices of the ear pattern by w_k for $k = 1, \ldots, 6$.

Body pattern. The body pattern is a rectangle with vertices $b_1, \ldots, b_4$. These vertices are shown in Figure 5.

Forming the union of the components. We put all pieces together as shown in Figure 5. The legs of the variable patterns are such that a guard at point w_1 sees all the legs of the variable patterns. We call the polygon obtained at this stage P'.

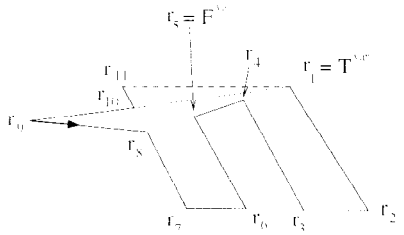

Fig. 4. Variable pattern.

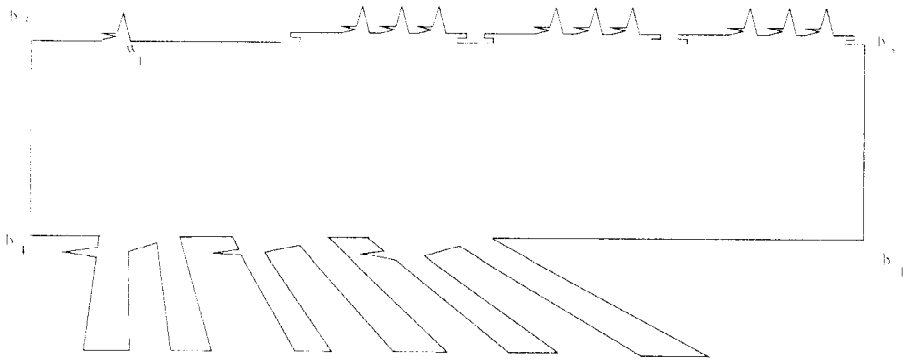

Fig. 5. Putting the pieces together.

Spike patterns. A spike pattern s is a triangle shaped polygon with some additional vertices on the edges. In the final polygon, there will be one spike pattern for each vertex T^{lit} and F^{lit}, which are of slightly different types. Figure 6(a) shows the type of spike patterns for vertices T^{lit}, which we call TRUE spike pattern; Figure 6(b) shows the type of spike patterns for vertices F^{lit}, which we call FALSE spike pattern. The spike pattern s is the polygon with vertices $s_1, \ldots, s_5, (s_6)$. We have the following collinearities:

– Vertices s_2, s_3, s_4 are collinear.
– Vertices $s_2, s_1, (s_6), s_5$ are collinear.

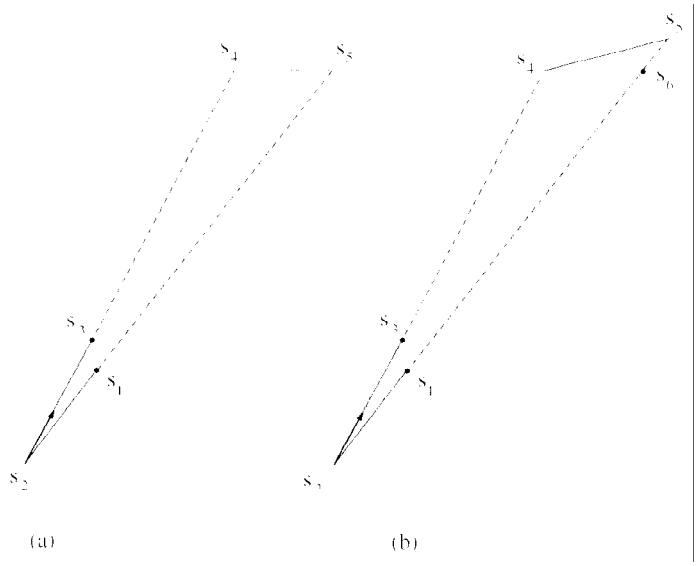

Fig. 6. (a) TRUE spike pattern; (b) FALSE spike pattern.

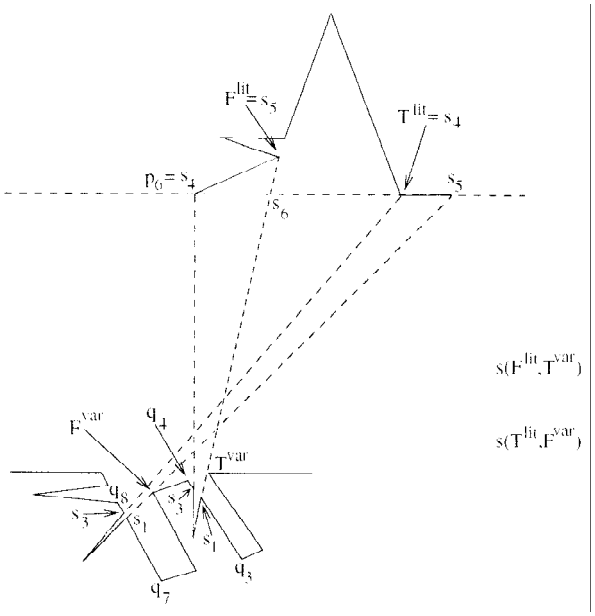

Fig. 7. Spikes patterns for a positive literal.

The tuple (s_2, s_3, s_1, s_3) is a distinguished tuple, if we change the definition in such a way that the edge from s_4 to s_5 is not transparent, as indicated in Figure 6 and in such a way that the view of a guard that is located on the extension of the line from s_2 through s_4 is blocked by the vertex s_4.

Adding spikes to the construction. We form the union of the spikes with the polygon P' as follows: We construct for each literal $l_j(c_i)$ in each clause two spike patterns (one TRUE and one FALSE spike pattern) as shown in Figures 7 and 8.

Figure 7 is for the case, when literal $l_j(c_i)$ is positive (i.e., $l_j(c_i) = x_k$, for some k). In this case we have a TRUE spike pattern $s(T^{\text{lit}}(l_j(c_i)), F^{\text{var}}(x_k))$, which connects vertex T^{lit} of the literal pattern $l_j(c_i)$ with vertex F^{var} of the variable pattern x_k, and a FALSE spike pattern $s(F^{\text{lit}}(l_j(c_i)), T^{\text{var}}(x_k))$, which connects vertex F^{lit} of the literal pattern $l_j(c_i)$ with vertex T^{var} of the variable pattern x_k.

Figure 8 is for the case when literal $l_j(c_i)$ is negative (i.e., $l_j(c_i) = \neg x_k$, for some k). In this case we have a TRUE spike pattern $s(T^{\text{lit}}(l_j(c_i)), T^{\text{var}}(x_k))$, which connects vertex T^{lit} of the literal pattern $l_j(c_i)$ with vertex T^{var} of the variable pattern x_k, and a FALSE spike pattern $s(F^{\text{lit}}(l_j(c_i)), F^{\text{var}}(x_k))$, which connects vertex F^{lit} of the literal pattern $l_j(c_i)$ with vertex F^{var} of the variable pattern x_k.

For each TRUE spike pattern s, we have the following, where p_6 and T^{lit} are vertices of the corresponding literal pattern:

- $s_4 = T^{\text{lit}}$.
- s_4, s_5, and p_6 are collinear.

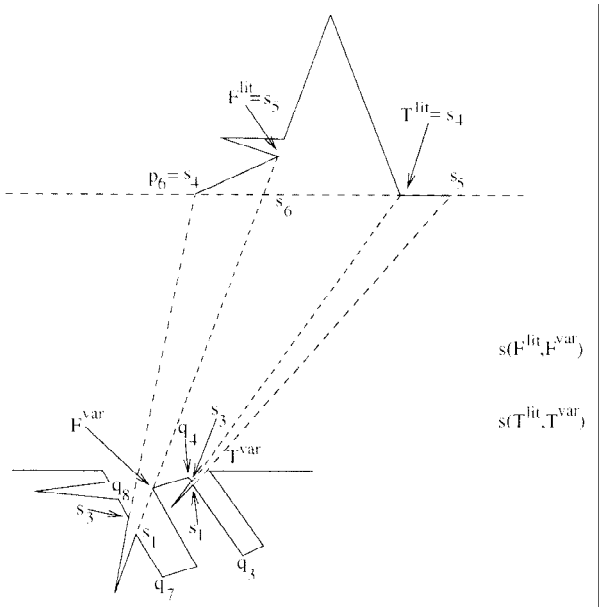

Fig. 8. Spike patterns for a negative literal.

For each FALSE spike pattern s, we have the following, where p_6 and T^{lit} are vertices of the corresponding literal pattern:

- $s_5 = F^{\text{lit}}$.
- $s_4 = p_6$.
- s_4, s_6, and T^{lit} are collinear.

For each spike pattern $s(T^{\text{lit}}, F^{\text{var}})$ or $s(F^{\text{lit}}, F^{\text{var}})$ we have the following collinearities, where q_7, q_8, and F^{var} are vertices of the corresponding variable pattern:

- s_1, F^{var}, s_5 are collinear.
- q_7, s_1, s_3, q_8 are collinear.

For each spike pattern $s(T^{\text{lit}}, T^{\text{var}})$ or $s(F^{\text{lit}}, T^{\text{var}})$ we have the following collinearities, where q_3, q_4, and T^{var} are vertices of the corresponding variable pattern:

- s_1, T^{var}, s_5 are collinear.
- q_3, s_1, s_3, q_4 are collinear.

As a result, we obtain the polygon P, which is the instance I' of PG.

Feasibility of the construction. Remember that, in order to see an arbitrarily small part of the edge from s_2 to s_3 including s_2 of each spike pattern s, a guard must lie in the interior or on the boundary of s, because (s_2, s_3) is a distinguished arrow.

In order to prove our inapproximability result for PG, we must ensure that the following holds:

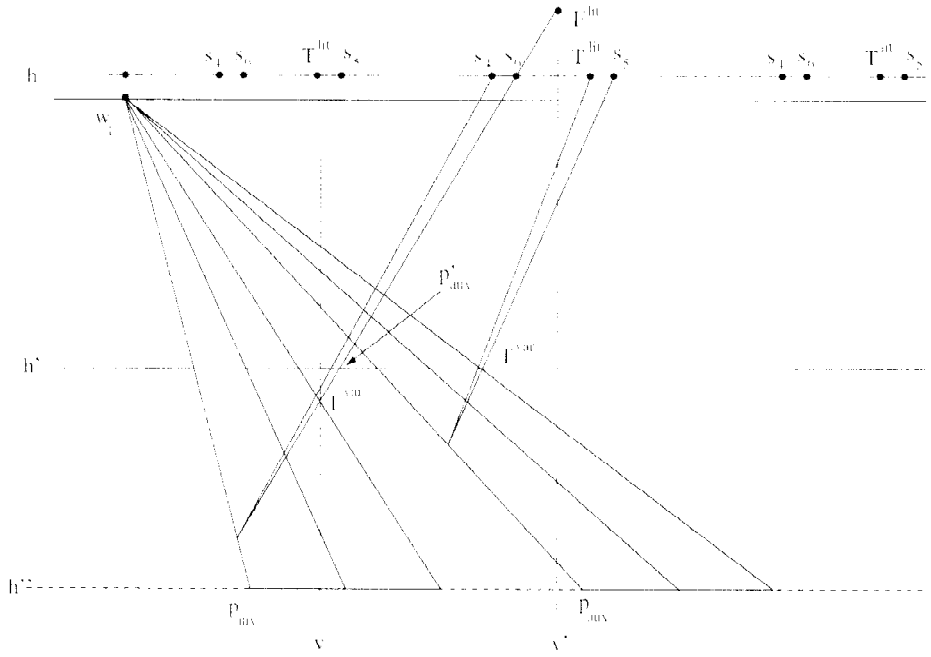

Fig. 9. Detailed construction.

LEMMA 1. *Instance I' of PG* (*i.e., the polygon P*) *can be constructed from the* 5-OCCURRENCE-MAX-3-SAT *instance I in such a way that no three spike patterns that connect literal patterns to three different legs* (*of the variable patterns*) *intersect in a common point.*

PROOF. We prove the lemma by giving a detailed description of how this can be achieved. An overview of the construction is given in Figure 9.

We start with the first literal pattern $l_1(c_1)$. First, fix vertex $s_5(T^{\text{lit}}(l_1(c_1)), \cdot)$ of the TRUE spike pattern on a horizontal line h. Then set vertex $T^{\text{lit}}(l_1(c_1))$ at a distance $a(I)$ to the left of $s_5(T^{\text{lit}}(l_1(c_1)), \cdot)$ on the horizontal line h. Fix vertex $s_6(F^{\text{lit}}(l_1(c_1)), \cdot)$ of the FALSE spike pattern at a constant distance to the left of $T^{\text{lit}}(l_1(c_1))$ on h, then set vertex $s_4(F^{\text{lit}}(l_1(c_1)), \cdot)$ at distance $a(I)$ to the left of $s_6(F^{\text{lit}}(l_1(c_1)), \cdot)$ on h. Then fix vertex $s_5(T^{\text{lit}}(l_2(c_1)), \cdot)$ of the TRUE spike pattern of the second literal pattern at constant distance to the left of $s_4(F^{\text{lit}}(l_1(c_1)), \cdot)$ on h and repeat the procedure for all literals.

Note that $a(I)$ depends on the instance (i.e., $a = a(I)$). Choose w_1 (of the ear pattern) at a constant distance to the left of the point $s_5(F^{\text{lit}}(l_3(c_m)), \cdot)$ (of the leftmost literal) and at distance $a'(I)$ below the line h.

Assume that the variable patterns for the variables $x_1, \ldots, x_{k-1}$ have already been constructed, that the vertices T^{var} of all of these variable patterns lie on the same horizontal line h', which is at constant distance from h, and that the vertices r_2, r_3, r_6, r_7 all lie on the same horizontal line h'', which is at distance $a''(I)$ from h'. We show how to

construct the next variable pattern for variable x_k. We determine point $T^{\text{var}}(x_k)$ of the variable pattern as follows:

Determine the rightmost of the (at most five) literal patterns that is a literal of x_k. We assume that it is a negative literal $l_j(c_i)$. (The case where it is a positive literal can be treated similarly.)

Set vertex $T^{\text{var}}(x_k)$ on the horizontal line h' to the left of all the variable patterns already constructed in such a way that there exist no areas where two spike patterns connecting literals with two different legs of variable patterns intersect, to the left or on the line from $T^{\text{var}}(x_k)$ to $s_5(T^{\text{lit}}(l_j(c_i)), T^{\text{var}}(x_k))$.

The intersection of the line h'' with the line from w_1 through T^{var} yields vertex r_2. Fix r_3 at a constant distance to the left of r_2 on h'' and fix some (auxiliary) point p_{aux} at a constant distance to the left of r_3 on h''. The intersection of the line from s_5 through T^{var} with the line from w_1 to p_{aux} yields vertex s_2; it yields vertex s_1 if intersected with the line from w_1 to r_3 . Intersecting the line from w_1 to p_{aux} with the line from T^{lit} to s_2 gives vertex s_3. Thus, we have constructed the TRUE leg with the first spike pattern. Now, construct all remaining spike patterns for the leg. Note that their vertices s_2 are strictly below vertex s_2 of the first spike pattern constructed. Also note that the distance $a(I)$ must be chosen small enough such that no two spike patterns intersect to the left of the line from r_3 to w_1.

We construct the FALSE leg in a similar way, however, we need an auxiliary point p'_{aux}, which is set on the horizontal line h' in such a way that there exist no areas where two spike patterns connecting literals with two different legs of variable patterns intersect, to the left or on the line from p'_{aux} to the vertex s_5 (or s_6) of the spike pattern that connects the rightmost literal pattern that represents a literal of the variable. Let v be a vertical line at some constant distance to the left of p'_{aux}. Vertex F^{var} is the intersection point of v with either the line from s_5 (or s_6) through p'_{aux} or the line from T^{var} through vertex s_3 of the topmost spike pattern in the TRUE leg, whichever is closer to the horizontal line h'. The remaining vertices of the leg and the spike patterns are then constructed as in the TRUE leg.

The variable pattern can now be completed by just observing the required collinearities.

Once we have constructed all variable patterns, we need to construct the literal patterns. For each literal, proceed as follows:

Let v' be a vertical line at some constant distance to the right of vertex s_6 of the FALSE spike pattern of the literal. Vertex F^{lit} is the intersection of v' with the line from s_1 through s_6 of the FALSE literal pattern. Constructing the remaining vertices of the literal pattern is straightforward by observing the required collinearities.

We complete the clause pattern in a straightforward manner, observing all collinearities and the requirement that a guard at a vertex F^{lit} of some literal pattern may not see the distinguished arrow of the corresponding clause pattern. Vertices q_1 and q_8 of each clause pattern are on the same horizontal line as w_1, which is at distance $a'(I)$ from line h. Note that $a'(I)$, therefore, must be chosen small enough such that the polygonal chains from q_1 to q_4 and from q_5 to q_8 do not intersect any spike patterns.

We complete the construction as indicated in Figure 5.

An analysis reveals that the coordinates of all points can be computed in polynomial time; some coordinates require a polynomial number of bits. The analysis is similar to the

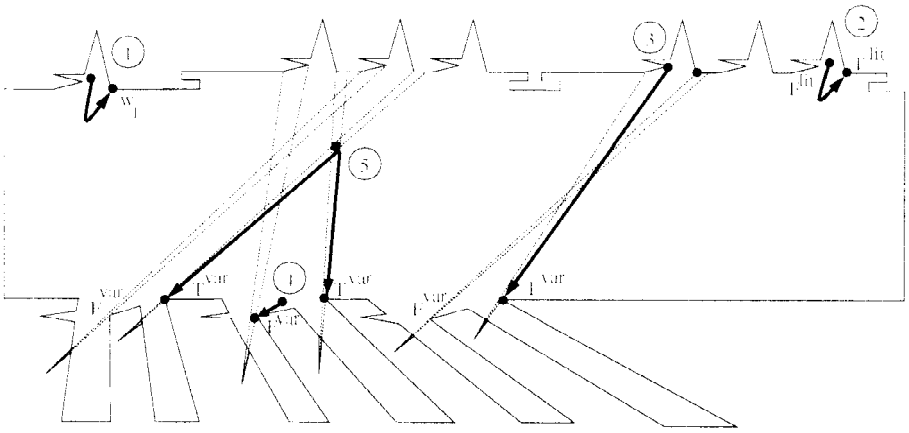

Fig. 10. Different types of guard moves.

analysis for PGH, which is given in full detail in Section 4. Therefore, the construction is polynomial in the size of the input. □

3.2. *Transformation of a Feasible Solution.* We describe how to obtain an assignment of the variables of the satisfiability instance (i.e., a solution of I), given a feasible solution of the corresponding PG instance I'. We move guards in such a way that the set of distinguished arrows that a guard sees changes in only one of two ways: The first possibility is that the set remains the same or contains some additional distinguished arrows. The second possibility is that some distinguished arrows are removed from the set, but then it is ensured that some other guards see the distinguished arrows that were removed. The guards are moved as follows (we have illustrated some of these movements in Figure 10):

- Determine which guard is inside the ear pattern $w_1, \ldots, w_6$ (see Figure 5) and move this guard to w_1 as indicated at ① in Figure 10.
- For each literal pattern, determine which guard sees the distinguished arrow (such a guard must be inside the literal pattern). If this guard is at vertex F^{lit}, then leave it there, otherwise, move it to vertex T^{lit} as indicated at ② in Figure 10.
- If there is a guard at both vertices T^{lit} and F^{lit} of the literal pattern, move the guard at F^{lit} along the edge of the FALSE spike pattern toward vertex s_2 to the vertex F^{var} or T^{var} of the corresponding variable pattern as indicated at ③ in Figure 10; note that the guard at T^{lit} does not move.
- If there is more than one guard at some vertex T^{lit} (F^{lit}), move all but one guard along the edge of the TRUE (FALSE) spike pattern toward vertex s_2 to the vertex F^{var} or T^{var} of the corresponding variable pattern.
- For each clause pattern, move any guard that sees the distinguished arrow of the clause pattern and lies in a TRUE spike pattern to the vertex T^{lit} that is on the boundary of the spike pattern.
- For each clause pattern, consider a guard g that sees the distinguished arrow of the clause pattern and lies in a FALSE spike pattern. If there already is a guard g' at vertex

F^{lit} of the spike pattern, then move the guard g to vertex T^{lit}, otherwise, move the guard g to vertex F^{lit}.
- For each variable pattern, move the guard that sees the distinguished arrow of the variable pattern to vertex T^{var}, if it also lies in a spike pattern that contains vertex T^{var}, and move it to point F^{var}, otherwise, as indicated at ④ in Figure 10.
- Move all guards that lie in a spike pattern but do not see the distinguished arrows of any literal or clause pattern to vertex T^{var} or F^{var} that is contained in the spike pattern.
- If a guard sees the distinguished arrows of two spike patterns that connect literals to two different legs of variable patterns, add a guard and move one guard each to the two vertices T^{var} or F^{var} of the variable patterns that lie on the boundary of the two spike patterns as indicated at ⑤ in Figure 10. (Note that because of Lemma 1, no guard can see the distinguished arrows of three spike patterns that belong to three different legs.)
- Guards that do not see any distinguished arrows are moved to any point T^{var} or F^{var} of any variable pattern, if there is no guard there already.

This procedure is iterated until all guards are at their final position. The solution obtained after moving and adding guards as indicated is still feasible. To see this, note that after this procedure there is exactly one guard in each literal pattern at either point F^{lit} or T^{lit}. In each clause pattern c_i there is at least one guard at either $T^{\text{lit}}(l_1(c_i))$, $T^{\text{lit}}(l_2(c_i))$, or $T^{\text{lit}}(l_3(c_i))$. Therefore, all literal and clause patterns are guarded. The remaining polygon (except for parts of the spike patterns) is guarded by the guard at point w_1 of the ear. Finally, the spike patterns are guarded, since all guards that saw the distinguished arrow of a spike pattern have been moved only within the spike pattern. Where such a guard saw two distinguished arrows of two spike patterns, we have added a guard.

We are now ready to set the truth values of the variables. For each variable pattern x_k, if there is a guard at point $F^{\text{var}}(x_k)$ and no guard at point $T^{\text{var}}(x_k)$, let x_k be false. If there is a guard at point $T^{\text{var}}(x_k)$ and no guard at point $F^{\text{var}}(x_k)$, let x_k be true. If there is a guard at both $T^{\text{var}}(x_k)$ and $F^{\text{var}}(x_k)$, then set x_k in such a way that a majority of the literals of x_k become true.

3.3. *Analysis of the Reduction.* We first prove two lemmas that will help us prove the APX-hardness of PG.

LEMMA 2. *If an instance of* 5-OCCURRENCE-MAX-3-SAT *with n variables and* $m \leq \frac{5}{3}n$ *clauses is satisfiable* (*i.e., all m clauses are satisfied*), *then there exists a feasible solution of the corresponding instance of PG with* $3m + n + 1$ *guards*.

PROOF. Fix any truth assignment of the variables that satisfies the 5-OCCURRENCE-MAX-3-SAT instance. Place one guard at w_1. For each variable x_k, place a guard at point $F^{\text{var}}(x_k)$ of the variable pattern if x_k is false. Place a guard at $T^{\text{var}}(x_k)$ if x_k is true. For each literal $l_j(c_i)$ in each clause, place a guard at point $T^{\text{lit}}(l_j(c_i))$ of the literal pattern if the literal is true. Place a guard at point $F^{\text{lit}}(l_j(c_i))$ if the literal is false. This solution is feasible and consists of $3m + n + 1$ guards. □

LEMMA 3. *If there exists an* $\varepsilon > 0$ *and a feasible solution of the PG instance* I' *with* $3m + n + 1 + \varepsilon m$ *guards, then there exists an assignment of the variables of the*

corresponding 5-OCCURRENCE-MAX-3-SAT *instance* I *that satisfies at least* $m(1-4\varepsilon)$ *clauses.*

PROOF. In the feasible solution of PG there must be at least one guard inside each literal pattern. There also must be at least one guard inside each variable pattern. Finally, one additional guard is needed at w_1.

Now, move the guards according to the transformation given in Section 3.2. At most εm guards see the distinguished pairs of two spike patterns belonging to different legs, since $3m$ guards are inside literal patterns, n guards are inside variable patterns and one guard is at w_1. Therefore, we have at most εm additional guards.

We now set the truth values of the variables according to the transformation. For at least $n - 2\varepsilon m$ variable patterns, there is only one guard at either $T^{\text{var}}(x_k)$ or $F^{\text{var}}(x_k)$. For at most $2\varepsilon m$ variable patterns, there is a guard at both points $T^{\text{var}}(x_k)$ and $F^{\text{var}}(x_k)$. When we set the truth value of each of these $2\varepsilon m$ variables, at most two clauses will be unsatisfied for each variable. Therefore, we have at most $4\varepsilon m$ unsatisfied clauses. □

Now, consider the promise problem of 5-OCCURRENCE-MAX-3-SAT, where we are given an instance of 5-OCCURRENCE-MAX-3-SAT, and we are promised that the instance is either satisfiable or at most $m(1-4\varepsilon)$ clauses are satisfiable by any assignment of the variables. The *NP*-hardness of this problem for small enough values of ε follows from the fact that 5-OCCURRENCE-MAX-3-SAT is *MAXSNP*-complete (see [21] and [2]).

By Lemma 2 and by the contraposition of Lemma 3, we obtain the following theorem.

THEOREM 1. *Let* I *be an instance of the promise problem of* 5-OCCURRENCE-MAX-3-SAT, *let* n *be the number of variables in* I, *and let* $m \leq \frac{5}{3}n$ *be the number of clauses in* I. *Let* $OPT(I)$ *denote the maximum number of satisfiable clauses* (*for any assignment*). *Furthermore, let* I' *be the corresponding instance of PG and let* $OPT(I')$ *denote the minimum number of guards needed to cover* I'. *Then the following hold*:

- *If* $OPT(I) = m$, *then* $OPT(I') \leq 3m + n + 1$.
- *If* $OPT(I) \leq m(1-4\varepsilon)$, *then* $OPT(I') \geq 3m + n + 1 + \varepsilon m$.

Theorem 1 shows that our reduction is gap-preserving (see [2]). It shows that the promise problem of PG with parameters $3m + n + 1$ and $3m + n + 1 + \varepsilon m$ is *NP*-hard. Note that $m \geq n/3$, since each variable appears as a literal at least once. Therefore, unless $NP = P$, no polynomial time approximation algorithm for PG can achieve an approximation ratio of

$$\frac{3m + n + 1 + \varepsilon m}{3m + n + 1} = 1 + \frac{\varepsilon}{3 + (n+1)/m} \geq 1 + \frac{\varepsilon}{3 + 3(n+1)/n} \geq 1 + \frac{\varepsilon}{7}.$$

Thus, we have the following result:

THEOREM 2. *PG is APX-hard.*

Our proof works as well for PGI. To see this note that if we are given a solution of the satisfiability instance and set the guards as indicated in Lemma 2, the guards see all

of the interior and the boundary of the polygon. If we are given a solution of the PGI instance and perform the transformation as given in Section 3.2, the guards still see all of the interior and the boundary of the polygon. Therefore, we have:

THEOREM 3. *PGI is APX-hard.*

3.4. *Inapproximability Results for VG, EG, VGI, and EGI.* The proof for the *APX*-hardness of VG can be copied from the corresponding proof for PG. Actually, we do not even need the property in the constructed polygon that no three spike pattern of three different legs intersect (see Lemma 1). Since there are no guards added in the transformation for VG, we would get a slightly bigger constant for the inapproximability of VG than the constant for PG, if we were interested in giving explicit constants.

Thus, we obtain:

THEOREM 4. *VG is APX-hard.*

Our proof carries over to VGI using the same arguments as for PGI. Therefore, we have:

THEOREM 5. *VGI is APX-hard.*

The proof for the *APX*-hardness of EG follows the lines of the corresponding proof for PG with some modifications. The literal pattern and the variable pattern are slightly different as shown in Figure 11. Note that F^{lit}, T^{lit}, F^{var}, and T^{var} are edges in the literal pattern.

The ideas of the proof for PG can now be applied here. Any solution of the EG instance contains at least $3m + n + 1$ guards. If we are given a solution of the EG instance we can adopt the transformation procedure described in Section 3.2. Therefore, we obtain:

THEOREM 6. *EG is APX-hard.*

The proof works as well for EGI. Therefore, we have:

THEOREM 7. *EGI is APX-hard.*

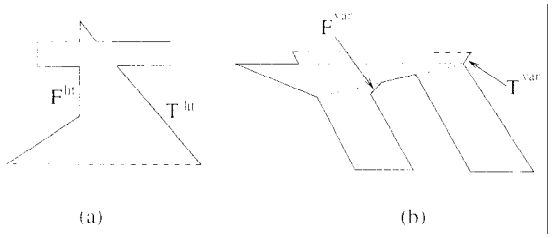

Fig. 11. (a) Literal pattern for EG; (b) variable pattern for EG.

4. Inapproximability Results for Guarding Polygons with Holes. In this section we propose a reduction from SET COVER to PGH, analyze it, and show that it is gap-preserving. We also show that our result carries over to the problems VGH, EGH, VGIH, EGIH, and PGIH.

4.1. *Construction of the Reduction.* As a first step toward our inapproximability result for PGH, we show how to construct an instance of PGH for every instance of SET COVER (SC). The construction contains a triangle-shaped pattern, called spike, for each element of the SC instance. All spikes lie on the lower segment of a large rectangle, which is "cut" into an upper and a lower part by a barrier that contains trapezoidal holes, through which a guard in the upper part, which corresponds to a set in the SC instance, can see the spikes in the lower part, which correspond exactly to those elements that are in the set.

We construct a polygon in the x–y plane; Figure 12 shows this construction. For each set s_i, $i = 1, \ldots, m$, place the point $((i-1)d', y_0)$ on the horizontal line $y = y_0$. This places a sequence of points from left to right, one point per set s_i for $i = 1, \ldots, m$, with d' a constant distance between two adjacent points. For ease of description, call the ith point s_i. For each element $e_i \in E$, on the horizontal line $y = 0$ place two points $(D_i, 0)$ and $(D_i', 0)$, with $D_i' = D_i + d$ for a positive constant d and $D_1 \geq 0$. Arrange the points from left to right for $i = 1, \ldots, n$, with distances $d_i = D_{i+1} - D_i'$ to be defined later. Call the points also D_i and D_i', for $i = 1, \ldots, n$.

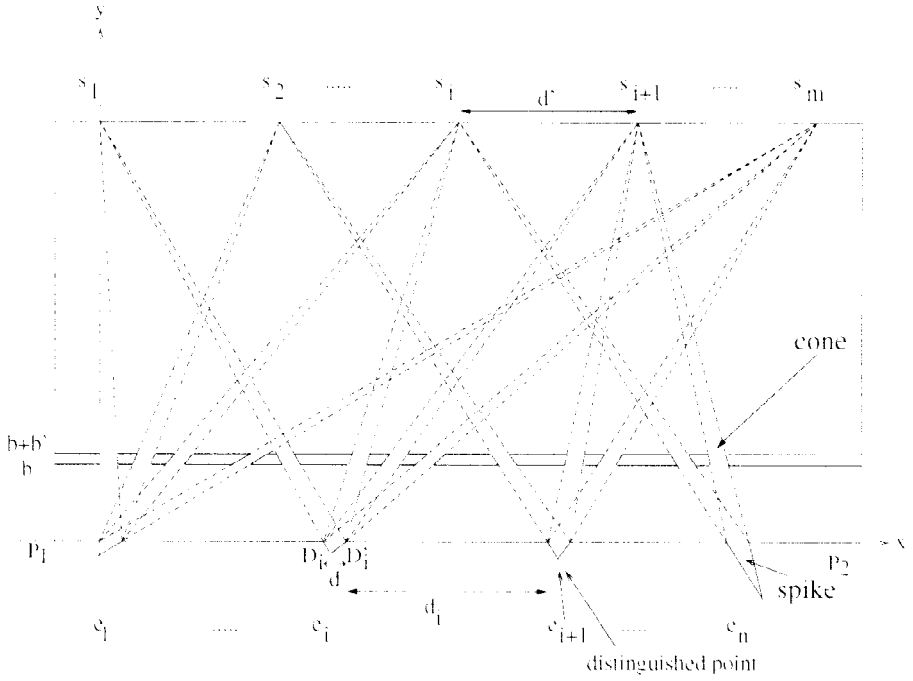

Fig. 12. Basic construction.

For every element e_i, draw a line g through s_j and D_i, where s_j is the first set of which e_i is a member. Also draw a line g' through s_l and D_i', where s_l is the last set of which e_i is a member.[3] For simplicity, let e_i also denote the intersection point of g and g'. Then draw line segments from every s_k that has e_i as a member to D_i and to D_i'.

Two lines connecting points D_i and D_i' with points s_j form a cone-like feature; the area between these two lines will therefore be called a *cone*. Call the triangle $D_i e_i D_i'$ a *spike*. The point e_i of each spike plays a special role and is therefore called the *distinguished point* of the spike.

We have only constructed one part of the polygon thus far: Among all the lines described, only the spikes and the line segments of the horizontal line $y = 0$ that are between adjacent spikes are part of the polygon boundary, all other lines merely help in the construction.

In our construction the guards of an optimum solution will have to be placed at or near the points s_j, therefore we need to make sure that a guard at s_j only sees the distinguished point e_i, if the element e_i is a member of the set s_j. This is achieved by introducing a "barrier" line at $y = b$, see Figure 12. Only line segments on the horizontal line $y = b$ that are outside the cones are part of the polygon boundary. We draw another barrier line with distance b' from the first barrier at $y = b + b'$. Define holes of the polygon by connecting endpoints of line segments of the two barrier lines that belong to the same cone-defining line. We call the area between the two lines at $y = b$ and $y = b + b'$ (including all holes) the *barrier*. Thus, the barrier contains a small part of all cones.

As a next step in the construction of the polygon, draw a vertical line segment at $x = -d''$, where d'' is a positive constant, from $y = 0$ to $y = y_0$. This line segment is part of the polygon boundary except for the segment between the two barrier lines.

Choose the coordinates (to be shown later) such that the rightmost spike is farther right than the rightmost set, i.e., $D_n' > s_m$ (for reasons of space, we violated this condition in Figure 12), and draw another vertical line segment from $y = 0$ to $y = y_0$ at $x = D_n' + d''$, again taking a detour at the barrier. The boundary lines of the polygon defined so far are shown as solid lines in Figure 12. It is important to note that the cones, drawn as dashed lines in the figures, are not part of the polygon boundary.

The thickness b' of the barrier is defined such that all segments of all holes except for those on the line $y = b + b'$ can be seen from two guards at $P_1 = (-d'', 0)$ and $P_2 = (D_n' + d'', 0)$. To achieve this, the thickness b' is determined by intersecting (for each pair of adjacent holes) a line from P_1 through the lower right corner (point G_1 in Figure 13) of the left hole (of the pair of adjacent holes) with a line from P_2 through the lower left corner (point G_2) of the right hole as shown in Figure 13. Now, the barrier line $y = b + b'$ is defined to go through the lowest of all these intersection points (point y_1 in Figure 13). (They are indeed all at the same height, by arguments with similar triangles.)

We set the parameters of the reduction as follows: Let d' and y_0 be arbitrary positive constants. Let d and b be positive constants as well, where $d = d'/4$ and $b = \frac{5}{12} y_0$. We let

$$b' = \frac{(35/12^2) y_0}{-4^{l-1} m^{l-1} + 2 \sum_{i=0}^{l-1} 4^i m^i + 2(d''/d) - \frac{19}{12}}$$

and $D_l = -4^{l-1} m^{l-1} d - d + 2d \sum_{i=0}^{l-1} 4^i m^i$ for $l = 1, \ldots, n$.

[3] We assume without loss of generality that each element is a member of at least two sets.

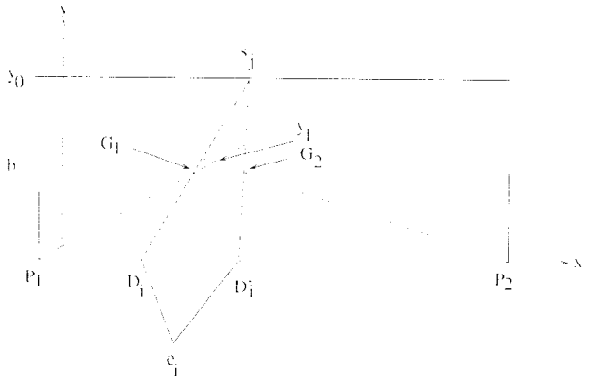

Fig. 13. Thickness of the barrier.

4.2. *Properties of the Reduction.* In order for the reduction to work, it is necessary that at no point does a guard see three or more distinguished points e_i unless there is a set s_j or a pair of sets s_j, s_l that (together) contain(s) all of the corresponding elements.

A guard that is placed at some point with y-value between 0 and $b + b'$, i.e., between the barrier and the spikes, sees at most one such distinguished point, provided the barrier is placed such that no cones of two different elements intersect in the area below the barrier and in the barrier.

In order to ensure that a guard that is placed at some point with y-value between $b + b'$ and y_0 does not see three or more distinguished points unless there is a set s_j or a pair of sets s_j, s_l that (together) contain(s) all of the corresponding elements, we introduce the notion of *extended cones* as shown in Figure 14. The extended cone is the area in the rectangle $D_i, D_i', s_j + a, s_j - a$. Point $s_j - a$ is defined as the intersection point of the line $y = y_0$ with the line from D_i' through the lower right corner of the left of the two holes which contain a part of the cone from set s_j and element e_i. Point $s_j + a$ is defined

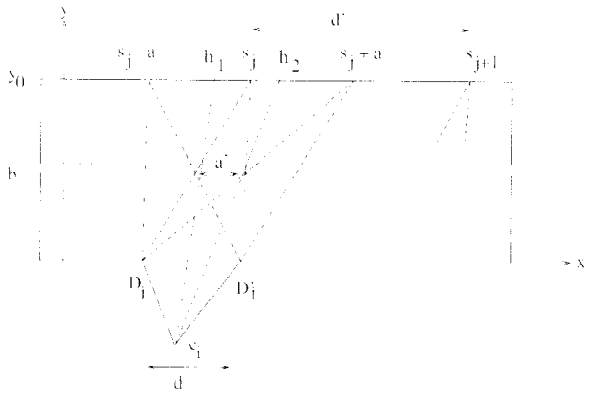

Fig. 14. Extended cones.

accordingly. It will be easy to see that points $s_j - a$ and $s_j + a$ are both at a constant distance a from point s_j (see the proof of Lemma 4).

For a guard between the two horizontal lines $y = b+b'$ and $y = y_0$, in order to see the distinguished point e_i, it must lie in the area of the triangle defined by the points h_1, h_2, and e_i (or, of course, in the corresponding triangle of any other point $s_{j'}$ with $e_i \in s_{j'}$). In order to keep the analysis simple, we argue with the extended cones rather than the triangles. If no three extended cones from three different elements and three different sets intersect in this area, then it is ensured that there exists a pair of setpoints such that each distinguished point that a guard in this area sees can also be seen from at least one of the setpoints of the pair. (It is, of course, also possible that a single setpoint sees all the distinguished points that a guard in this area sees).

A guard that is placed at some point with y-value less than 0, sees at most one distinguished point, if it is ensured that no two spikes intersect.

Thus, we need to prove the following:

- No three extended cones from different elements and sets intersect.
- The barrier is such that all intersections of cones from the same element e_i are below b (to ensure that the view of the points s_j is blocked appropriately) and such that all intersections of cones from different elements are above $b + b'$ and such that all of the barrier except for the line segments at $y = b + b'$ can be seen from at least one of two guards at P_1 and P_2.
- No two spikes intersect.

No three extended cones from different elements and sets intersect

LEMMA 4. *For $e_l \in s_{l'}$, let*

$$D_l \;\geq\; \max\left(\frac{s_{i'} - s_{l'}}{s_{i'} - s_{j'} - 2a}(D_j + d - D_i) + D_i + d\right),$$

where the maximum is taken over all $e_i \in s_{i'}$ and $e_j \in s_{j'}$, for which $i < j < l$ and $l' < j' < i'$ holds. Then the three extended cones from e_l to $s_{l'}$, from e_i to $s_{i'}$ and from e_j to $s_{j'}$, with $i < j < l$, do not have a common intersection point.

PROOF. Assume that the positions of the elements, i.e., the values D_v, have been set for all $v < l$ such that no three extended cones (connecting three different sets with three different elements) intersect. We show how to set D_l such that no three extended cones intersect; see Figure 15. Let S be an intersection point with the maximum y-value among the two extended cones connecting the elements e_i and e_j with the (different) points $s_{j'}$ and $s_{i'}$.

In order to ensure that our construction is feasible, S must lie in the area between y_0 and the barrier. Let S_y be the y-value of S. Then $S_y < y_0$. To see this, note that this is equivalent to saying that $s_{j'} + a < s_{i'} - a$ (see Figures 14 and 15), which is a weaker condition than $s_{j'} + a < s_{j'+1} - a$. Now, $s_{j'} + a < s_{j'+1} - a$ is equivalent to $2a < d'$. We express a as a function of y_0, b, and d using the similarity of triangles. Note that $a'/d = (y_0 - b)/y_0$ and $b/y_0 = a'/a$. Thus, we get $a = ((1 - b)/b)d$. Using this result in $2a < d'$, we obtain

$$b \;>\; \frac{2}{d'/d + 2} y_0,$$

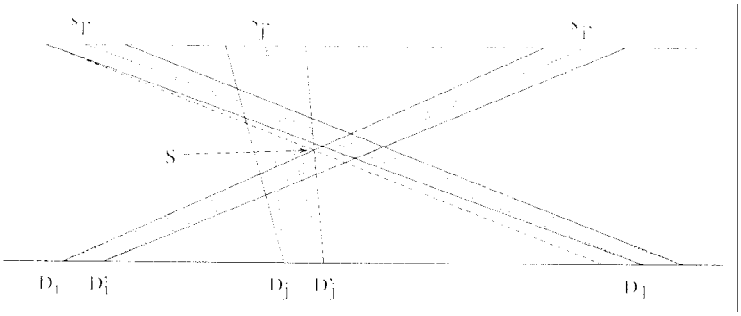

Fig. 15. Intersection of three extended cones.

which is equivalent to $b > \frac{1}{3}y_0$, since $d = d'/4$. This inequality for b is satisfied, since $b = \frac{5}{12}y_0 > \frac{1}{3}y_0$.

For each set $s_{l'}$ of which e_l is a member, draw a line through S, determine where it intersects the line $y = 0$ and let $D^S_{l,l'}$ be the x-value of this intersection point. Let $D^S_l = \max_{l'} D^S_{l,l'}$ be the maximum x-value of all intersection points defined this way. For any pair of extended cones in "inverse position" to the left of e_l, with which an extended cone at e_l forms a "triple inversion," compute the corresponding D^S_l and let $D^{\max}_l$ be the maximum D^S_l. Finally, we let $D_l = D^{\max}_l + d$ to ensure that no three extended cones have one common intersection point at some point S. Figure 15 shows the situation for an intersection and explains the notation.

The point S is the intersection point of the lines g_1 from $s_{i'} - a$ to D_i and g_2 from $s_{j'} + a$ to D'_j.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1\colon \quad (1-t)\begin{pmatrix} s_{i'} - a \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i \\ 0 \end{pmatrix},$$

$$g_2\colon \quad (1-t)\begin{pmatrix} s_{j'} + a \\ y_0 \end{pmatrix} + t\begin{pmatrix} D'_j \\ 0 \end{pmatrix}.$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0, \\ (1-t_1)(s_{i'} - a) + t_1 D_i &= (1-t_2)(s_{j'} + a) + t_2 D'_j. \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains, for t_1,

$$t_1 = \frac{s_{i'} - s_{j'} - 2a}{D'_j - D_i + s_{i'} - s_{j'} - 2a}.$$

We express S as

$$S = \begin{pmatrix} (1-t_1)(s_{i'} - a) + t_1 D_i \\ y_0(1-t_1) \end{pmatrix}.$$

Let g_3 be the line from $s_{l'} - a$ to S with $t \in \mathbb{R}$ as parameter:

$$g_3\colon \quad (1-t)\begin{pmatrix} s_{l'} - a \\ y_0 \end{pmatrix} + t\begin{pmatrix} (1-t_1)(s_{i'} - a) + t_1 D_i \\ y_0(1-t_1) \end{pmatrix}.$$

The intersection of g_3 and $y = 0$ is characterized by parameter t_3:

$$\begin{aligned} (1-t_3)y_0 + t_3(1-t_1)y_0 &= 0, \\ t_3 &= \frac{1}{t_1}. \end{aligned}$$

We let $D^S_{l,l'}$ be the corresponding x-value:

$$\begin{aligned} D^S_{l,l'} &= \left(1 - \frac{1}{t_1}\right)(s_{l'} - a) + \frac{1}{t_1}(1-t_1)(s_{i'} - a) + \frac{1}{t_1}t_1 D_i \\ &= \left(1 - \frac{1}{t_1}\right)(s_{l'} - s_{i'}) + D_i \\ &= \frac{s_{i'} - s_{l'}}{s_{i'} - s_{j'} - 2a}(D'_j - D_i) + D_i \\ &= \frac{s_{i'} - s_{l'}}{s_{i'} - s_{j'} - 2a}(D_j + d - D_i) + D_i. \end{aligned}$$

The lemma follows. □

Lemma 4 implies

$$\begin{aligned} &\max\left(\frac{s_{i'} - s_{l'}}{s_{i'} - s_{j'} - 2a}(D_j + d - D_i) + D_i + d\right) \\ &\qquad \le \max\left(\frac{md'}{d' - 2a}(D_j + d) + d\right) \qquad (\forall j < l) \\ &\qquad \le 4m(D_{l-1} + d) + d, \end{aligned}$$

where we have used $a = ((1-b)/b)d = \frac{7}{5}d$ and $d' = 4d$ in the last step. Now, let $D_l = 4m(D_{l-1} + d) + d$. It is easy to see that this is consistent with our definition of D_l, since

$$-4^{l-1}m^{l-1}d - d + 2d\sum_{i=0}^{l-1} 4^i m^i \;=\; 4m\left(\left(-4^{l-2}m^{l-2}d - d + 2d\sum_{i=0}^{l-2} 4^i m^i\right) + d\right) + d.$$

The barrier is in good position

LEMMA 5. *Any two cones that belong to the same element e_i intersect only at points with y-values at most $y_0(d/(d+d'))$.*

PROOF. Let e_i be a member of s_j and s_l and $s_j < s_l$. The intersection point of the lines g_1 from s_j to D_i' and g_2 from s_l to D_i is the point in the intersection area of the two cones that has the largest y-value. Let this value be y_c.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1: \quad (1-t)\begin{pmatrix} s_j \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i' \\ 0 \end{pmatrix},$$

$$g_2: \quad (1-t)\begin{pmatrix} s_l \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i \\ 0 \end{pmatrix}.$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0, \\ (1-t_1)s_j + tD_i' &= (1-t_2)s_l + tD_i. \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains, for t_1,

$$t_1 = \frac{s_l - s_j}{D_i' - D_i + s_l - s_j}.$$

Since $D_i' - D_i = d$ and since $s_l - s_j \geq d'$, we get

$$\begin{aligned} y_c &= y_0 \frac{d}{d + s_l - s_j} \\ &\leq y_0 \frac{d}{d + d'}. \end{aligned}$$

□

LEMMA 6. *Any two cones that belong to elements e_i, e_j, respectively, with $i < j$, intersect only at points with y-values at least $y_0(d_i/(d_i + md'))$.*

PROOF. Let e_i be a member of $s_{i'}$ and let e_j be a member of $s_{j'}$, also let $D_i < D_j$ and $s_{j'} < s_{i'}$. Exactly then, the corresponding two cones intersect.

The intersection point of the lines g_1 from $s_{j'}$ to D_j and g_2 from $s_{i'}$ to D_i' is the point in the intersection area of the two cones with the minimum y-value. Let this value be y_c.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1: \quad (1-t)\begin{pmatrix} s_{j'} \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_j \\ 0 \end{pmatrix},$$

$$g_2: \quad (1-t)\begin{pmatrix} s_{i'} \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_i' \\ 0 \end{pmatrix}.$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0, \\ (1-t_1)s_{j'} + tD_j &= (1-t_2)s_{i'} + t_2 D_i'. \end{aligned}$$

The first equation leads to $t_1 = t_2$ and one obtains, for t_1,

$$t_1 = \frac{s_{i'} - s_{j'}}{D_j - D'_i + s_{i'} - s_{j'}}.$$

Since $D_j - D'_i \geq d_i$ and since $s_{i'} - s_{j'} \leq md'$, we get

$$\begin{aligned} y_c &= y_0 \frac{D_j - D'_i}{D_j - D'_i + s_{i'} - s_{j'}} \\ &\geq y_0 \frac{d_i}{d_i + md'}. \end{aligned}$$

□

LEMMA 7. *Let*

$$b' = \frac{bd(y_0 - b)}{y_0(p_2 - p_1) - d(y_0 - b)},$$

where p_1 and p_2 are the x-values of the points P_1 and P_2. Then all of the barrier including the segments of the cones except for the segments at $y = b + b'$ can be seen from the two guards at P_1 and P_2.

PROOF. Let $e_i \in s_j$ and let G_1 and G_2 be the two points where this cone intersects with the barrier line $y = b$ (see Figure 13). We need to find an expression for y_1, which is the y-value of the intersection point of the two lines from P_1 to G_1 and from P_2 to G_2.

We find an expression for the point G_1 by calculating the intersection of the lines from s_j to D_i and $y = b$ and obtain

$$G_1 = \begin{pmatrix} (b/y_0)(s_j - D_i) + D_i \\ b \end{pmatrix}.$$

We find an expression for the point G_2 by calculating the intersection of the lines from s_j to $D_i + d$ and $y = b$ and obtain

$$G_2 = \begin{pmatrix} (b/y_0)(s_j - D_i - d) + D_i + d \\ b \end{pmatrix}.$$

Now, we find the intersection point of the lines from P_1 to G_1 and from P_2 to G_2:

$$\begin{aligned} &(1 - t_1)\begin{pmatrix} p_1 \\ 0 \end{pmatrix} + t_1 \begin{pmatrix} (b/y_0)(s_j - D_i) + D_i \\ b \end{pmatrix} \\ &\quad = (1 - t_2)\begin{pmatrix} p_2 \\ 0 \end{pmatrix} + t_2 \begin{pmatrix} (b/y_0)(s_j - D_i - d) + D_i + d \\ b \end{pmatrix}. \end{aligned}$$

Again, $t_1 = t_2$ and we obtain

$$t_1 = \frac{p_1 - p_2}{d - bd/y_0 + p_1 - p_2}.$$

Therefore,

$$y_1 = bt_1.$$

y_1 does not depend on D_i, therefore we let $b' = y_1 - b = b(t_1 - 1)$:

$$b' = \frac{bd(y_0 - b)}{y_0(p_2 - p_1) - d(y_0 - b)}. \qquad \square$$

If we substitute $b = \frac{5}{12}y_0$ and $p_2 - p_1 = -4^{n-1}m^{n-1}d - d + 2d\sum_{i=0}^{n-1} 4^i m^i + d'' - (-d'') = -4^{n-1}m^{n-1}d - d + 2d\sum_{i=0}^{n-1} 4^i m^i + 2d''$ in the equation for b', we obtain

$$b' = \frac{(35/12^2)y_0}{-4^{n-1}m^{n-1}2\sum_{i=0}^{n-1} 4^i m^i + 2(d''/d) - \frac{19}{12}}.$$

A simple calculation shows that $b' < y_0/12$, if $m \geq 2$ and $n \geq 2$, which must be the case since there were no intersections otherwise.

Because $d = d'/4$ and because of Lemma 5, any two cones from the same element intersect only at points with a y-value at most $\frac{1}{5}y_0$ which is less than b. Because $d_i \geq md'$ for all d_i and because of Lemma 6, any two cones from different elements intersect only at points with a y-value at least $\frac{1}{2}y_0$, which is at most $b + b'$.

Spikes of two elements do not intersect

LEMMA 8. *The spikes of any two elements do not intersect.*

PROOF. Let s_i be the first and let s_j be the last set that e_l is a member of. Obviously, $s_i < s_j$. The intersection point of the lines g_1 from s_i through D_l and g_2 from s_j through D'_l is the point I_l. Let the x-value of this point be x_l. Note that $x_l > D_l$.

These two lines can be expressed with parameter $t \in \mathbb{R}$:

$$g_1\colon \quad (1-t)\begin{pmatrix} s_i \\ y_0 \end{pmatrix} + t\begin{pmatrix} D_l \\ 0 \end{pmatrix},$$
$$g_2\colon \quad (1-t)\begin{pmatrix} s_j \\ y_0 \end{pmatrix} + t\begin{pmatrix} D'_l \\ 0 \end{pmatrix}.$$

The intersection is characterized by parameters t_1 and t_2 for g_1 and g_2:

$$\begin{aligned} (1-t_1)y_0 &= (1-t_2)y_0, \\ (1-t_1)s_i + t_1 D_l &= (1-t_2)s_j + t_2 D'_l. \end{aligned}$$

The first equation leads to $t_1 = t_2$ and with $D'_l = D_l + d$ one obtains, for t_1,

$$t_1 = \frac{s_i - s_j}{d + s_i - s_j}.$$

Thus, we obtain

$$\begin{aligned} x_c &= s_i \frac{d}{d + s_i - s_j} + D_l \frac{s_i - s_j}{d + s_i - s_j} \\ &\leq D_l \frac{s_i - s_j}{d + s_i - s_j} \end{aligned}$$

$$\begin{aligned} &\leq D_l \frac{-d'}{d-d'} \\ &= D_l \frac{-4d}{d-4d} \\ &= \tfrac{4}{3} D_l, \end{aligned}$$

where the second but last step is due to $d = d'/4$. Since $D_{l+1} = 4m(D_l + d) + d$ and since we can assume that $m \geq 1$, the lemma follows. □

4.3. *Transformation of the Solution.* Given a solution of the PGH-instance, i.e., the coordinates of r guards $g_1, \ldots, g_r$, proceed as follows to obtain a solution for the SC-instance:

For each guard g_i determine the set h_i of elements e_j of which the guard g_i sees the corresponding distinguished point e_j.

Since no three extended cones from three different elements and three different sets intersect in the area above $y = b + b'$ by our construction, there exists a pair of sets (s_k, s_l) for each guard g_i such that $h_i \subseteq s_k \cup s_l$. Determine such a pair of sets for each guard g_i and add the sets to the solution of the SC-instance.

4.4. *The Reduction is Polynomial.* Note that d, d', y_0, h, b are all constants in our reduction. The values for b' and for all D_i are computable in polynomial time and can be expressed with $O(n \log m)$ bits.

Therefore, the construction of the polygon can be done in time polynomial in the size of the input SC-instance, since it only produces a polynomial number of points that can each be computed in polynomial time and each take at most $O(n \log m)$ bits to be expressed.

It is obvious that the transformation of the solution runs in polynomial time, since it only involves determining whether two points see each other and finding pairs of sets for a polynomial number of guards. (Note that if the number of guards exceeds n, the solution is trivial.)

4.5. *An Inapproximability Result for PGH.* In order to prove a strong inapproximability result, we need the following:

DEFINITION 19. The RESTRICTED SET COVER (RSC) problem consist of all SET COVER instances that have the property that the number of sets m is less than or equal to the number of elements n, i.e., $m \leq n$.

LEMMA 9. RESTRICTED SET COVER *cannot be approximated by any polynomial time algorithm with an approximation ratio of* $(1 - \varepsilon) \ln n$ *for any* $\varepsilon > 0$, *unless NP* $\subseteq$ *TIME*$(n^{O(\log \log n)})$.

PROOF. We know that DOMINATING SET cannot be approximated with an approximation ratio of $(1 - \varepsilon) \ln n$ for any $\varepsilon > 0$, unless *NP* $\subseteq$ *TIME*$(n^{O(\log \log n)})$, where n is the number of vertices in the graph [4]. Consider the following reduction from DOM-

INATING SET to RESTRICTED SET COVER: given a graph $G = (V, E)$ with $n := |V|$, which is an instance of DOMINATING SET, we construct a RESTRICTED SET COVER instance by letting the vertices of G be elements and by forming a set for each vertex that contains the vertex itself as well as its neighbors. The RESTRICTED SET COVER instance thus obtained contains n elements and n sets. This is clearly a gap-preserving reduction, since each feasible solution of the RESTRICTED SET COVER instance directly corresponds to a feasible solution (of the same size) of the DOMINATING SET instance. □

We now consider the reduction to be from RSC to PGH (rather than from SC to PGH).

LEMMA 10. *Consider the promise problem of RSC (for any $\varepsilon > 0$), where it is promised that the optimum solution OPT is either less than or equal to c or greater than $c(1-\varepsilon)\ln n$ with c, n, and OPT depending on the instance I. This problem is NP-hard unless NP $\subseteq$ TIME$(n^{O(\log\log n)})$ (see the notion of quasi-NP-hardness in* [2]*). Then we have for the optimum value OPT$'$ of the corresponding PGH-instance I', that OPT$'$ is either less than or equal to $c+2$ or greater than $((c+2)/12)(1-\varepsilon)\ln|I'|$. More formally,*

$$OPT \le c \quad \Rightarrow \quad OPT' \le c+2, \tag{1}$$

$$OPT > c(1-\varepsilon)\ln n \quad \Rightarrow \quad OPT' > \frac{c+2}{12}(1-\varepsilon)\ln|I'|. \tag{2}$$

PROOF. The implication in (1) is trivial, since, given a solution of the RSC-instance I of size c, we position a guard at each point s_j in the corresponding PGH-instance I', if the set s_j is in the solution of I, and we position two additional guards at points P_1 and P_2 in I', which see the barrier from below.

We prove the contraposition of (2), i.e.,

$$OPT' \le \frac{c+2}{12}(1-\varepsilon)\ln|I'| \quad \Rightarrow \quad OPT \le c(1-\varepsilon)\ln n.$$

Observe that if we are given a solution of I' with k guards, we can obtain a solution of I with at most $2k$ sets by performing the procedure described in Section 4.3. Therefore,

$$OPT \le 2\frac{c+2}{12}(1-\varepsilon)\ln|I'| \tag{3}$$

$$\le 2\frac{c+2}{12}(1-\varepsilon)\ln n^3 \tag{4}$$

$$\le 2\cdot 3\frac{2c}{12}(1-\varepsilon)\ln n \tag{5}$$

$$\le c(1-\varepsilon)\ln n, \tag{6}$$

where we used $|I'| \le n^3$ to get (4), which is true because the polygon of I' consists of n spikes and less than $nm \le n^2$ holes (see the definition of RSC). Therefore, the polygon consists of less than $k(n^2+n)$ points, where k is a small constant. Therefore, $|I'| \le n^3$ for n large enough. We used $2c \ge c+2$ to get to (5). □

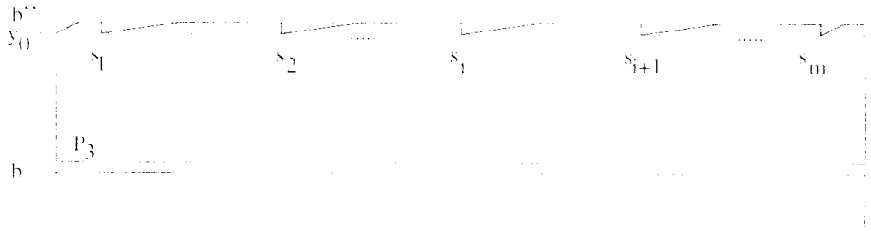

Fig. 16. Polygon for VG and EG.

Lemma 10 completes the proof of Theorem 8.

THEOREM 8. *PGH cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of the polygon vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

4.6. *Inapproximability Results for VGH and EGH.* A slight modification of the polygon as indicated in Figure 16, where $b'' = y_0 + b'$, allows us to prove the corresponding theorems for VGH and EGH.

THEOREM 9. *VGH cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of polygon vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. The proof is almost identical to the proof for PGH, except that instead of two additional guards at P_1 and P_2 we have a third additional guard at P_3 (see Figure 16). This additional guard means that we need to replace $c+2$ by $c+3$ in the proof of Lemma 10. In addition, we get a slightly stronger condition, namely $2c \geq c+3$, to obtain the inequality at (5). □

THEOREM 10. *EGH cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of polygon vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. The proof is almost identical to the proof for PGH with the additional information from the proof of Theorem 9. Note that in the case of EG all guards are edges. The proofs carry over effortlessly. □

4.7. *Inapproximability Results for PGIH, VGIH, and EGIH*

THEOREM 11. *PGIH, VGIH, and EGIH cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of polygon vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. All proofs which lead to Theorems 8–10 carry over. Note that for a lemma corresponding to Lemma 10, which is the most crucial part in the proof, we can still, virtually without change, prove (1) and (2). □

5. An Application: Guarding Terrains. We prove inapproximability results for several terrain guarding problems by proposing reductions from RESTRICTED SET COVER, all of which are based on the reduction proposed for PGH.

THEOREM 12. *PGT cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of terrain vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. We reduce RESTRICTED SET COVER to PGT. In a first step, we construct the same polygon with holes as constructed in the corresponding reduction for PGH. We then triangulate this polygon arbitrarily, and construct a terrain, by letting the interior of the polygon have height 0 and the exterior (including the holes in the barrier) have height h', for a positive constant h'. To make the terrain finite, we cut off the exterior of the polygon with a generous bounding box that is triangulated as well. The terrain we obtain has vertical walls, which is for reasons of simplicity only. The terrain can easily be modified to have steep, but not vertical, walls.

The proof then carries on just as for PGH. The setpoints s_j are assumed to be at height h', as are points P_1 and P_2; the distinguished points e_i, however, are at height 0. □

THEOREM 13. *VGT cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of terrain vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. Adopt the proof of Theorem 12 for PGT with the modifications of the constructed polygon as indicated in Theorem 9 for VGH. □

THEOREM 14. *FHT cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of terrain vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. Proceed as in the proof for Theorem 12, and let h, the fixed height, where the guards can be placed, be equal to h', the height of the exterior of the polygon. □

THEOREM 15. *PGTR, VGTR, and FHTR cannot be approximated by a polynomial time algorithm with an approximation ratio of* $((1-\varepsilon)/12)\ln n$ *for any* $\varepsilon > 0$, *where n is the number of terrain vertices, unless NP* $\subseteq$ *TIME*$(n^{O(\log\log n)})$.

PROOF. Proceed as in the proofs for Theorems 12–14. However, triangulate the polygon in such a way that each spike is triangulated into a single triangle. Note that the spikes take over the role of the distinguished points. A solution for RSC with k sets

can still be easily transformed into a solution of the terrain guarding instance with k guards. Furthermore, a solution of the terrain guarding instance with k guards can still be transformed into a solution of the RSC instance with at most $2k$ sets, because three different cones from three different elements and three different sets do not intersect in the area above the barrier, because this is a weaker condition than the corresponding condition with extended cones. □

6. Approximability Results. Surprisingly few approximation algorithms are known for art gallery problems and terrain guarding problems.

It is known, however, that VGI, EGI, VGIH, and EGIH are approximable with a ratio of $O(\log n)$, where n is the number of polygon vertices [13]. The corresponding approximation algorithms divide the interior of the polygons into "basic" triangles that are either completely visible or invisible from any vertex- or edge-guard. The problem is then transformed into an instance of SET COVER, which can be approximated with a logarithmic ratio by a greedy algorithm, which consists of recursively adding to the solution the set that covers a maximum number of elements not yet covered and achieves an approximation ratio of $\ln n + 1$ [14]. These algorithms can be easily modified to work for VG, EG, VGH, and EGH as well.

No sophisticated approximation algorithms are known for PG, PGH, PGI, and PGIH, except for a restricted version of PGI [1]. In fact, it is not even known if the corresponding decision problems are in *NP*. A trivial approximation algorithm for PG, PGH, PGI, and PGIH simply returns all n vertices of the polygon as a (feasible) solution. This algorithm achieves an approximation ratio of n, because there is at least one guard needed in a feasible solution. Note that this ratio might be improved slightly for PGI by applying an algorithm that places $\lfloor n/3 \rfloor$ guards that together see all of the interior of the polygon (see [25] for details); this could be done similarly for PGIH with another algorithm (see [25] for details), but the approximation ratio remains $O(n)$.

For terrain guarding problems, we have the following results.

THEOREM 16. *FHTR can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$.

PROOF. In order to prove the theorem, we construct an SC-instance for a given FHTR-instance as follows: Each triangle is an element of the SC-instance. For each triangle determine the area on the plane $z = h$ from where the triangle is fully visible. This area is a polygon of descriptional complexity $O(n^2)$, that can be computed in time $O(n^4)$ by interpreting the points of the polygon as special points of an arrangement. At each point, where two of these polygons intersect, determine which triangles are visible from this point and define the set of visible triangles as one set for SC. There are $O(n^6)$ such intersections. Now solve the SC-instance approximately, by applying the greedy algorithm for SC. The solution obtained is not more than $\ln n + 1$ times larger than the optimum solution for SC [14].

To see that this reduction is approximation-ratio preserving, consider that the n polygons partition the plane $z = h$ into cells. Observe that the set of visible triangles is the same throughout the area of a cell. On the boundary of the cell, however, a few more

triangles might be visible since the boundary may be part of the visibility area of another triangle. Therefore, any solution of the FHTR-instance can be transformed to a solution of the SC-instance by moving guards that are in the interior of a cell to an appropriate intersection point on the boundary of the cell. □

THEOREM 17. *VGTR can be approximated by a polynomial time algorithm with a ratio of* $O(\log n)$.

PROOF. Similarly as in the proof of Theorem 16, we construct an SC-instance for a given VGTR-instance. Each triangle in the terrain is an element of the SC-instance. We determine at each vertex in the terrain, which triangles are completely visible from a guard at the vertex and define the set of visible triangles as one set for SC. We then apply the greedy algorithm to the SC-instance and obtain a solution which is not more than a logarithmic factor away from the optimum. □

No sophisticated approximation algorithms are known for PGT, PGTR, VGT, and FHT. A trivial approximation algorithm for VGT, PGT, and PGTR simply puts a guard at each vertex of the terrain, thus achieving an approximation ratio of n, since at least one guard is always needed. A trivial approximation algorithm for FHT with approximation ratio n places a guard above each vertex of the terrain at height h. Again, the approximation ratios could be improved by a constant factor, since there exists an algorithm [5] that always places $\lfloor 3n/5 \rfloor$ (vertex-)guards on a terrain that together see all of the terrain. For FHT, we could also reduce the approximation ratio to $n/2$ by determining whether height h is large enough such that the whole terrain can be seen from one single guard at some point at height h. The position of such a guard can be computed in linear time using linear programming (mentioned in [26] as the problem of computing the *lowest watchtower*). An approximation algorithm for FTH could return the position of such a guard and if no such guard exists, it would proceed as in the trivial algorithm. However, the approximation ratios remain $O(n)$ for all four problems.

7. Conclusion. Table 1 gives an overview of the known results for the problems studied. The entries in the "upper bound" column are the smallest approximation ratios for the corresponding problems, which are achieved by known polynomial time algorithms. The entries in the "lower bound" column are the largest approximation ratios known, which cannot be achieved by polynomial time algorithms. The entries in the "unless" column show what the existence of polynomial time algorithms with better ratios than the ones given in the "lower bound" column would imply.

All our logarithmic inapproximablity results can actually also be proved under the weaker assumption that $NP \neq P$, however, with a slightly smaller factor than $1-\varepsilon$, since DOMINATING SET and therefore also RESTRICTED SET COVER cannot be approximated with an approximation ratio of $c \ln n$ for some $c > 0$ [3], [4], [22].

Table 1 shows that our inapproximability results are optimum up to constant factors for VGH, EGH, VGIH, EGIH, VGTR, and FHTR.

Thus, a fair amount of work remains to be done. One important issue is to find nontrivial approximation algorithms with significantly better approximation ratios than

Table 1. Summary of results.

Problem	Upper bound	Lower bound	Unless
VG	$O(\log n)$ [13]	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
EG	$O(\log n)$ [13]	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
PG	$O(n)$	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
VGI	$O(\log n)$ [13]	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
EGI	$O(\log n)$ [13]	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
PGI	$O(n)$	$1+\varepsilon$ for some $\varepsilon > 0$	$NP = P$
VGH	$O(\log n)$ [13]	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
EGH	$O(\log n)$ [13]	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
PGH	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
VGIH	$O(\log n)$ [13]	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
EGIH	$O(\log n)$ [13]	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
PGIH	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
VGT	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
PGT	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
FHT	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
VGTR	$O(\log n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
PGTR	$O(n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$
FHTR	$O(\log n)$	$\frac{1-\varepsilon}{12} \ln n$ for any $\varepsilon > 0$	$NP \subseteq TIME(n^{O(\log\log n)})$

$O(n)$ for the problems, for which only approximation algorithms with ratio $O(n)$ are known. Of course, one could also try to improve the corresponding inapproximability results.

Another problem is either to prove that VG, EG, VGI, and VGI cannot be approximated with some logarithmic ratio or to propose approximation algorithms for these problems with constant ratio.

Acknowledgement. The authors thank the anonymous referees for helpful suggestions based on an earlier version of this paper.

References

[1] A. Aggarwal, S. Ghosh, and R. Shyamasundar; Computational complexity of restricted polygon decompositions; in *Computational Morphology* (ed. G. Toussaint), pp. 1–11; North-Holland, Amsterdam, 1988.

[2] S. Arora and C. Lund; Hardness of approximations; in *Approximation Algorithms for NP-Hard Problems* (ed. D. Hochbaum), pp. 399–446; PWS, Boston, MA, 1996.

[3] S. Arora and M. Sudan; Improved low-degree testing and its applications; *Proc*. 29*th ACM Symposium on the Theory of Computing*, pp. 485–495, 1997.

[4] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi; A list of NP optimization problems; in *Complexity and Approximation*: *Combinatorial Optimization Problems and Their Approximability Properties*, pp. 367–470; Springer-Verlag, Berlin, 1999; also available in an online version at http://www.nada.kth.se/theory/compendium/.

[5] P. Bose, T. Shermer, G. Toussaint, and B. Zhu; Guarding polyhedral terrains; in *Computational Geometry* 7, pp. 173–185; Elsevier Science, Amsterdam, 1997.

[6] D. P. Bovet and P. Crescenzi; *Introduction to the Theory of Complexity*; Prentice-Hall, Englewood Cliffs, NJ, 1993.

[7] P. Crescenzi and L. Trevisan; On approximation scheme preserving reducibility and its applications, *Proc*. 14*th Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, pp. 330–341; Lecture Notes in Computer Science, Vol. 880; Springer-Verlag, Berlin, 1994.

[8] J. C. Culberson and R. A. Reckhow; Covering polygons is hard; *Proc*. 29*th Symposium on Foundations of Computer Science*, pp. 601–605, 1988.

[9] S. Eidenbenz; *Inapproximability Results for Guarding Polygons without Holes*, pp. 427–436; Lecture Notes in Computer Science, Vol. 1533; Springer-Verlag, Berlin, 1998.

[10] S. Eidenbenz, C. Stamm, and P. Widmayer; Inapproximability of some art gallery problems; *Proc*. 10*th Canadian Conference on Computational Geometry*, pp. 64–65, 1998.

[11] S. Eidenbenz, C. Stamm, and P. Widmayer; *Positioning Guards at Fixed Height above a Terrain – An Optimum Inapproximability Result*, pp. 187–198; Lecture Notes in Computer Science, Vol. 1461; Springer-Verlag, Berlin, 1998.

[12] U. Feige; A threshold of $\ln n$ for approximating set cover; *Journal of the ACM*, Vol. 45 No. 4, pp. 634–652, 1998; a preliminary version appeared in *Proc*. 28*th ACM Symposium on the Theory of Computing*, pp. 314–318, 1996.

[13] S. Ghosh; Approximation algorithms for art gallery problems; *Proc*. *Canadian Information Processing Society Congress*, 1987.

[14] D. Johnson; Approximation algorithms for combinatorial problems; *Journal of Computer and System Sciences*, Vol. 9, pp. 256–278, 1974.

[15] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani; On syntactic versus computational views of approximability; *Proc*. 35*th Annual IEEE Symposium on Foundations of Computer Science*, pp. 819–830, 1994.

[16] M. van Kreveld; Digital elevation models and TIN algorithms; in *Algorithmic Foundations of Geographic Information Systems* (ed. van Kreveld et al.), pp. 37–78; Lecture Notes in Computer Science, Vol. 1340; Springer-Verlag, Berlin, 1997.

[17] D. T. Lee and A. K. Lin; Computational complexity of art gallery problems; *IEEE Transactions on Information Theory*, Vol. 32, pp. 276–282, 1986.

[18] B. Nilsson; Guarding Art Galleries - Methods for Mobile Guards; Doctoral thesis, Department of Computer Science, Lund University, 1994.

[19] J. O'Rourke; *Art Gallery Theorems and Algorithms*; Oxford University Press, New York, 1987.

[20] J. O'Rourke and K. J. Supowit; Some *NP*-hard polygon decomposition problems; *IEEE Transactions on Information Theory*, Vol. 29, No. 2, pp. 181–190, 1983.

[21] C. H. Papadimitriou and M. Yannakakis; Optimization, approximation, and complexity classes; *Proc*. 20*th ACM Symposium on the Theory of Computing*, pp. 229–234, 1988.

[22] R. Raz and S. Safra; A sub-constant error-probability low-degree test, and a sub-constant error-probability *PCP* characterization of *NP*; *Proc*. 29*th ACM Symposium on the Theory of Computing*, pp. 475–484, 1997.

[23] T. Shermer; Recent results in art galleries; *Proceedings of the IEEE*, Vol. 80, pp. 1384–1399, 1992.

[24] L. Trevisan; Reductions and (Non-)Approximability, Doctoral thesis, 1997.

[25] J. Urrutia; Art gallery and illumination problems; in *Handbook on Computational Geometry* (ed. J.-R. Sack and J. Urrutia), pp. 973–1027; North-Holland, Amsterdam, 2000.

[26] B. Zhu; Computing the shortest watchtower of a polyhedral terrain in $O(n \log n)$ time; in *Computational Geometry* 8, pp. 181–193; Elsevier Science, Amsterdam, 1997.